

NAME

`vwrays` - compute rays for a given picture or view

SYNOPSIS

`vwrays [-i -u -f{a|f|d} -c rept | -d] { view opts .. | picture [zbuf] }`

DESCRIPTION

Vwrays takes a picture or view specification and computes the ray origin and direction corresponding to each pixel in the image. This information may then be passed to *rtrace(1)* to perform other calculations. If a given pixel has no corresponding ray (because it is outside the legal view boundaries), then six zero values are sent instead.

The `-i` option may be used to specify desired pixel positions on the standard input rather than generating all the pixels for a given view. If the `-u` option is also given, output will be unbuffered.

The `-f` option may be used to set the record format to something other than the default ASCII. Using raw float or double records for example can reduce the time requirements of transferring and interpreting information in *rtrace*.

The `-c` option repeats each pixel the given number of times (default is 1). This is most useful when sending rays to *rcontrib(1)* with the same `-c` setting, providing a much faster way to average pixels over image sets. The `-pj` option should be used to jitter sample positions in most cases.

View options may be any combination of standard view parameters described in the *rpict(1)* manual page, including input from a view file with the `-vf` option. Additionally, the target X and Y dimensions may be specified with `-x` and `-y` options, and the pixel aspect ratio may be given with `-pa`. The default dimensions are 512x512, with a pixel aspect ratio of 1.0. Just as in *rpict*, the X or the Y dimension will be reduced if necessary to best match the specified pixel aspect ratio, unless this ratio is set to zero. The `-pj` option may be used to jitter samples. The default value of 0 turns off ray jittering.

If the `-d` option is given, then *vwrays* just prints the computed image dimensions, which are based on the view aspect and the pixel aspect ratio just described. The `-ld` switch will also be printed, with `-ld+` if the view file has an aft clipping plane, and `-ld-` otherwise. This is useful for passing options to the *rtrace* command line. (See below.)

If the view contains an aft clipping plane (`-va` option), then the magnitudes of the ray directions will equal the maximum distance for each pixel, which will be interpreted correctly by *rtrace* with the `-ld+` option. Note that this option should not be given unless there is an aft clipping plane, since the ray direction vectors will be normalized otherwise, which would produce a uniform clipping distance of 1.

If a picture is given on the command line rather than a set of view options, then the view and image dimensions are taken from the picture file, and the reported ray origins and directions will match the center of each pixel in the picture (plus optional jitter).

If a depth buffer file is given as well, then *vwrays* computes the intersection point of each pixel ray (equal to the ray origin plus the depth times the ray direction), and reports this instead of the ray origin. The reported ray direction will also be reversed. The interpretation of this data is an image of origins and directions for light rays leaving the scene surfaces to strike each pixel.

EXAMPLES

To compute the ray intersection points and returned directions corresponding to a picture and its depth buffer:

```
vwrays scene_v2.hdr scene_v2.zbf > scene_v2.pts
```

To determine what the dimensions of a given view would be:

```
vwrays -d -vf myview.vf -x 2048 -y 2048
```

To generate a RADIANCE picture using *rtrace* instead of *rpict*:

```
vwrays -ff -vf view1.vf -x 1024 -y 1024 | rtrace 'vwrays -d -vf view1.vf -x 1024 -y 1024' -ffc scene.oct > view1.hdr
```

AUTHOR

Greg Ward Larson

ACKNOWLEDGMENT

This work was supported by Silicon Graphics, Inc.

BUGS

Although *vwrays* can reproduce any pixel ordering (i.e., any image orientation) when given a rendered picture, it will only produce standard scanline-ordered rays when given a set of view parameters.

SEE ALSO

rcalc(1), *rpict(1)*, *rcontrib(1)*, *rtrace(1)*